FB.RU\_INVOKE\_RUN

Детектор определяет ошибку, в ходе которой метод явно вызывает run() для объекта. В общем, классы реализуют интерфейс Runnable, потому что их метод run() будет вызываться в новом потоке, и в этом случае Thread.start() является подходящим методом для вызова.

Таблица №1 – Основные характеристики детектора

|  |  |
| --- | --- |
| Категория критичности | MAJOR |
| Надежность | Unknown |
| Источник | SpotBugs |
| Тип ошибки | Ошибка разработчика |
| CWE | CWE572 |
| CVE | — |
| Возможные последствия | Явный вызов run() вместо start() приводит к тому, что метод выполняется в текущем потоке, а не в новом. Это нарушает ожидаемую многопоточность и может привести к проблемам с параллелизмом, блокировкам и некорректному выполнению программы. |
| Комментарий | Если требуется выполнение в отдельном потоке, необходимо использовать Thread.start(), а не run(). Это гарантирует, что код внутри run() будет выполнен в новом потоке, как и предполагает использование Runnable. |

Таблица №2 – Примеры срабатывания детектора и устранения уязвимости

|  |  |
| --- | --- |
| Некорректное написание | Корректное написание |
| public static void incorrectTest() {  class Task implements Runnable {  @Override  public void run() {  System.out.println(Thread.currentThread().getName());  }  }  Task task = new Task();  // вызывается в текущем потоке, а не в новом  task.run();  } | public static void correctTest() {  class Task implements Runnable {  @Override  public void run() {  System.out.println(Thread.currentThread().getName());  }  }  Thread thread = new Thread(new Task());  // создается новый поток  thread.start();  } |